PROGRAMA DE RESIDÊNCIA EM TECNOLOGIA DA INFORMAÇÃO
TRIBUNAL DE CONTAS DO ESTADO DO RIO GRANDE DO NORTE

EDITAL 013/2019 – PROVA DE CONHECIMENTOS ESPECÍFICOS
ÁREA DE CONCENTRAÇÃO 1: DESENVOLVIMENTO DE SOFTWARE
16 / 06 / 2019

Identificação do Candidato	
Nome completo:	
CPF:	**Assinatura:**

Leia com atenção as seguintes instruções:

1. Aguarde a autorização do(s) fiscal(is) para poder iniciar a Prova.
2. Não esqueça de colocar seu **nome completo** (preferencialmente em letras maiúsculas) e de assinar o campo acima.
3. Este Caderno de Prova, com páginas numeradas de 1 a 16, é constituído de **30 (trinta) questões** de múltipla escolha, cada uma com quatro alternativas. Verifique se o Caderno de Prova está completo, sem falhas de impressão ou problemas que comprometam sua leitura. Caso necessário, solicite imediatamente ao(s) fiscal(is) a substituição do Caderno de Prova completo.
4. Confira se este Caderno de Prova corresponde à área de concentração para a qual foi inscrito. Caso haja alguma divergência, notifique imediatamente o(s) fiscal(is).
5. Leia com atenção o enunciado das questões antes de responde-las.
6. Cada questão possui **apenas uma** alternativa correta. Você deverá marcar a resposta que julgar correta usando caneta esferográfica de tinta na cor azul ou preta no local correspondente à respectiva questão. A interpretação das questões faz parte da Prova, de modo que não será permitida qualquer tipo de pergunta ou explicação ao(s) fiscal(is).
7. Não serão computadas questões sem marcação de resposta ou que contenham mais de uma marcação, marcação rasurada ou emendada.
8. Tenha cuidado ao manusear este Caderno de Prova, evitando rasuras, pois ele **não será substituído** por esse motivo. Também não é permitido destacar quaisquer das folhas que compõem este Caderno de Prova.
9. O tempo máximo para resolução desta Prova é de **2 (duas) horas**, para o qual **não haverá prorrogação**. Transcorrido esse tempo, o Caderno de Prova será recolhido pelo(s) fiscal(is).
10. Terminada a realização da Prova, este Caderno de Prova deverá ser **obrigatoriamente** entregue ao(s) fiscal(is) antes de se retirar da sala de realização.

QUESTÕES

1. Programas escritos em C# operam na presença do .NET Framework, que é uma tecnologia cujos elementos fundamentais são:
a) um interpretador universal e um conjunto unificado de tipos de dados.
b) um ambiente comum de tempo de execução (*runtime*) e uma biblioteca de classes.
c) um interpretador universal e um conjunto de serviços Web.
c) um ambiente comum de tempo de execução (*runtime*) e um ambiente comum de desenvolvimento.

2. Assinale a alternativa que apresenta, corretamente, o conceito do paradigma de Programação Orientada a Objetos que promove reutilização de software.
a) Herança
b) Sobrecarga de métodos
c) Abstração de dados
d) Polimorfismo

3. Com relação aos relacionamentos possíveis em Diagramas de Classes na UML (*Unified Modeling Language*), considere as afirmativas a seguir:

I. Uma agregação é uma forma mais forte de relacionamento de composição.
II. Uma parte pode pertencer a somente um todo de cada vez.
III. Somente uma classe no relacionamento pode representar o todo.
IV. As partes no relacionamento de composição só existem enquanto o todo existir.

Assinale a alternativa correta.
a) Somente as afirmativas I e IV são corretas.
b) Somente as afirmativas I e II são corretas
c) Somente as alternativas II, III e IV são corretas.
d) Somente as alternativas I, II e III são corretas.

4. Assinale a alternativa que completa de forma correta a seguinte descrição acerca de uma categoria de teste de software.

> *O teste _____ desconhece o conteúdo do código fonte. Nesse tipo de teste, a entidade a ser testada é tratada como uma espécie de caixa-preta: são fornecidos dados de entrada e o resultado é comparado com aquele esperado e previamente conhecido.*

a) de desempenho.
b) funcional.
c) de integração.
d) unitário.

5. Acerca de Arquitetura Orientada a Serviços (SOA), é incorreto afirmar:
a) SOA é um estilo arquitetural que preconiza que aplicações sejam fundamentalmente construídas através de serviços.
b) Em SOA, existem basicamente dois tipos de agentes de *software* trocando mensagens entre si, os provedores de serviços e os clientes que fazem uso de tais serviços.
c) SOA não é suficiente para promover interoperabilidade entre serviços, principalmente quando se tem aplicações desenvolvidas com diferentes tecnologias e plataformas e elas precisam comunicar-se umas com as outras.
d) SOA permite combinar serviços para fornecer funcionalidades agregadas, de mais alto nível.

6. Marque a alternativa correta a respeito de padrões de projeto.
a) Cada padrão de projeto oferece código-fonte que mostra como utilizar bibliotecas de classes diretamente na aplicação.
b) Cada padrão de projeto busca resolver um problema recorrente de projeto de *software*, oferecendo uma solução concreta dentro de um contexto específico.
c) Todo padrão de projeto permite flexibilizar um determinado aspecto da aplicação.
d) Padrões de projeto promovem o reuso de código de *frameworks* existentes.

7. No que se refere à linguagem HTML:
I. O elemento `<head>` é usado para a descrição do título do documento e pode ser subdividido em `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` e `<h6>`.
II. O elemento `<head>` é usado para a descrição de informações complementares sobre o documento, como, por exemplo, código de caracteres do documento e língua na qual ele está escrito.
III. O elemento `<head>` é o local para colocar o cabeçalho de uma página no *layout* em HTML 5.

Dentre as afirmações anteriores:
a) apenas II está correta
b) apenas III está correta
c) I e II estão corretas
d) I e III estão corretas

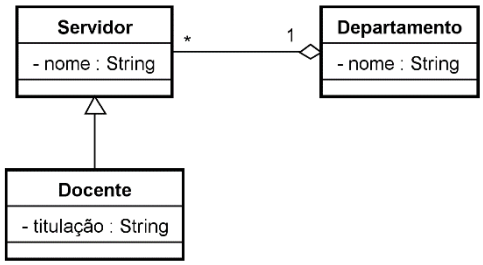
8. Importante conceito do paradigma de Programação Orientada a Objetos, o encapsulamento de dados tem por objetivo ocultar detalhes de implementação de um determinado módulo. Em linguagens de programação que seguem esse paradigma, tais como Java, esse ocultamento é obtido fazendo com que todos os membros (atributos e métodos) em uma classe tenham um nível particular de visibilidade com relação às suas subclasses e às classes que acessam esses membros. No que se refere aos níveis de visibilidade, assinale a alternativa correta:
a) Um membro público é visível a qualquer classe que acessa esse membro bem como à subclasse da classe à qual ele pertence.
b) Um membro protegido é visível somente à classe à qual ele pertence, mas não às suas subclasses ou às classes que o acessam.
c) Um membro privado é visível somente às subclasses da classe à qual ele pertence.
d) Um método público pode acessar somente atributos públicos declarados na classe à qual ele pertence.

9. Analise as seguintes afirmativas acerca da estrutura de aplicações na plataforma Android:

I. Uma *Task* corresponde a uma unidade de interação do usuário ou uma unidade de execução.
II. Um *Intent* representa uma descrição abstrata de uma operação que uma atividade requer que outra desempenhe.
III. Uma *Activity* representa uma cadeia de atividades que podem se estender por mais de um aplicativo.

Está(ão) correta(s) a(s) afirmativa(s):
a) II, apenas.
b) I e II, apenas.
c) I e III, apenas.
d) I, II e III.

10. Considere o seguinte Diagrama UML de Classes:



Com base no que está representado nesse diagrama, assinale a alternativa incorreta:
a) Todo docente está associado a um departamento.
b) Todo departamento tem ao menos um servidor.
c) Um departamento pode ter nenhum servidor associado.
d) Um departamento pode ter nenhum docente associado.

11. Com o passar dos anos, as aplicações corporativas evoluíram em sua arquitetura, saindo de um modelo monolítico executado em computadores de grande porte para um modelo em duas camadas (*two-tier*) cliente-servidor e então para um modelo contendo no mínimo três camadas (*three-tier*). Essas camadas são:
a) visualização, lógica e negócio
b) apresentação, negócio e acesso a dados
c) domínio, negócio e acesso a dados
d) sistemas, processos e bancos de dados

12. Considere o seguinte código fonte na linguagem de programação C#:

```
using System;
class Teste {
   public static void Main() {
      string[] nomes = { "Ana", "Pedro", "Maria" };
      for (int i = 0; i <= 2; i++) {
         ____________________________________________
      }
   }
}
```

Assinale a alternativa que preenche corretamente a lacuna no código fonte anterior.
a) System.out.println("Nome " + i + " = " + nomes[i]);
b) Console.WriteLine("Nome i = {nomes[i]}");
c) Console.WriteLine("Nome {0} = {1}", i, nomes[i]);
d) Console.WriteLine("Nome {%d, 0} = {%s, 1}", i, nomes[i]);

13. Um código fonte na linguagem de programação JavaScript é interpretado a partir de arquivos HTML que são carregadas em um navegador. Isso significa que, para que seja possível utilizar JavaScript em páginas Web, é necessário integrar o código fonte em JavaScript com o restante do código fonte em HTML. Nesse contexto, assinale V (verdadeiro) ou F (falso) às seguintes afirmações:

( ) Há duas formas de incluir código fonte em JavaScript em um documento HTML: embutindo o código no documento ou carregando o código a partir de um arquivo separado.
( ) Para embutir um código fonte em JavaScript em um arquivo HTML, é necessário utilizar a *tag* <script>, colocar o código fonte e fechar a *tag* com </script>.
( ) Outra forma de carregar código fonte em JavaScript em uma página Web é carregando um arquivo que contém esse código.
( ) Existe apenas uma forma de incluir código fonte JavaScript em um documento HTML, que é embutindo o código neste.
( ) Para incluir código fonte em JavaScript em um documento HTML, basta substituir a *tag* <html> pela *tag* <script> e mudar o nome do arquivo para a extensão .js.

Marque a alternativa que contém a sequência correta:
a) V, V, V, F, F
b) V, F, V, V, F
c) F, V, F, V, V
d) F, V, F, F, V

14. A UML (*Unified Modeling Language*) é uma linguagem visual de modelagem que pode ser utilizada para visualizar, especificar, construir e documentar artefatos relacionados a um *software*. Em relação aos diferentes diagramas que essa notação provê, é correto afirmar:

a) A UML 2.0 divide os diagramas em duas categorias básicas, a saber, diagramas estruturais e diagramas comportamentais. O Diagrama de Componentes é um diagrama comportamental que representa a topologia física do sistema, bem como os vários componentes de *software* de um sistema e suas dependências.

b) O Diagrama de Máquina de Estados permite visualizar um fluxo ou processo de negócio. Ele é especialmente útil para detalhar um caso de uso que descreve um fluxo complexo envolvendo muitas partes e ações concorrentes.

c) O Diagrama de Casos de Uso apresenta as funcionalidades externamente observáveis do sistema e os elementos externos com os quais ele interage. Nesse diagrama, um elemento externo que interage com o sistema é chamado de ator, que pode representar, por exemplo, pessoas, outros sistemas e equipamentos.

d) Um Modelo de Domínio, ilustrado como um conjunto de Diagramas de Classes, é uma representação de classes conceituais do mundo real e as restrições inerentes à tecnologia a ser utilizada na solução. É importante constarem nesse modelo os atributos e operações de cada classe.

15. Na definição de estilos com CSS (*Cascading Style Sheet*), é possível utilizar elementos, classes e identificadores. Nesse sentido, considere o seguinte trecho de código CSS com a definição de alguns estilos aplicáveis a elementos de uma página Web:

```
h1 .center {
  text-align: center;
  font-weight: bold;
}

#p1 {
  text-align: center;
  font-weight: bold;
}
```

Qual alternativa corresponde à utilização correta de estilos sobre os elementos de uma página Web?

a) <h1 id="center">Olá, mundo!</h1>

b) <h1 id="p1">Olá, mundo!</h1>

c) <p class="center">Olá, mundo!</h1>

d) <p class="p1">Olá, mundo!</h1>

16. Analise os seguintes códigos fonte escritos na linguagem de programação C++:

Código fonte 1:

```
#include <iostream>
using std::cout;
using std::endl;

int main() {
  int mat[2][2] = {{1, 2}, {3, 4}};
  for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
      cout << mat[i][j] << " ";
    }
  }
```

Código fonte 2:

```
#include <iostream>
using std::cout;

int main() {
  int mat[2][2] = {{1, 2}, {3, 4}};
  int *p = &mat[0][0];
  for (int i = 0; i < 4; i++) {
    cout << *(p+i) << " ";
  }
  return 0;
}
```

Assinale a alternativa que descreve corretamente o comportamento observado durante a execução do programa resultante da compilação desses códigos fonte:

a) O programa referente ao código fonte 1 imprimirá os valores da matriz *mat* e o referente ao código fonte 2 indicará um erro na inicialização do ponteiro *p*.
b) O programa referente ao código fonte 1 imprimirá os valores da matriz *mat* e o referente ao código fonte 2 indicará um erro no laço de repetição.
c) O programa referente ao código fonte 1 imprimirá os valores da matriz *mat* e o referente ao código fonte 2 imprimirá valores desconhecidos alocados na memória.
d) Ambos os programas terão o mesmo resultado impresso na saída padrão.

17. Padrões arquiteturais expressam formas de organizar os elementos que podem compor a arquitetura de um *software*, inclusive podendo auxiliar na definição dessa arquitetura pelo fato de exporem quando podem ser utilizados e documentarem suas respectivas vantagens e desvantagens. Associe as colunas, relacionando os padrões arquiteturais aos cenários em que podem ser utilizados.

Padrões

1. Cliente-servidor
2. Tubos e filtros
3. Camadas
4. *Model–View–Controller*
5. Repositório

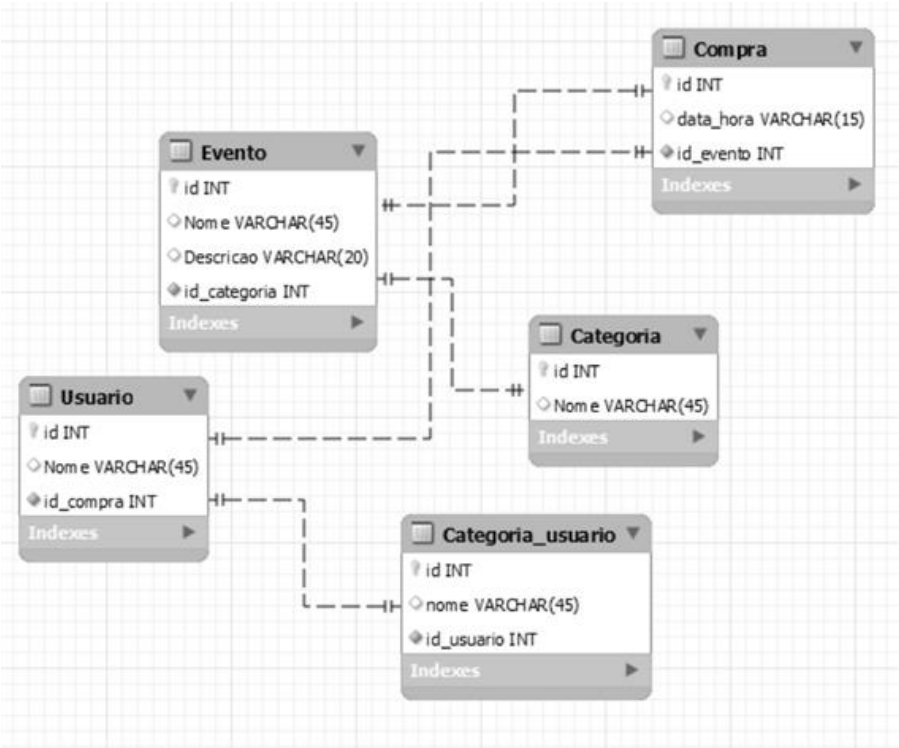
Cenários

( ) Quando há necessidade de manter uma gerência centralizada de todos os dados, de modo que estes sejam acessíveis a todos os componentes do sistema
( ) Em aplicações que envolvem a entrada de dados que são processados em etapas separadas, nas quais os dados fluem de um componente para outro para processamento
( ) Quando há possibilidade de incorporar novos requisitos não funcionais (tais como distribuição, segurança, persistência, etc.) de modo a minimizar modificações no restante do sistema em razão da integração desse novo requisito
( ) Quando há necessidade que os dados sejam mantidos de maneira independente de sua apresentação, de modo que possam existir diversas maneiras de visualizar e interagir com os dados
( ) Quando os dados compartilhados precisam ser acessados a partir de vários locais

A sequência correta dessa associação é
a) 5, 2, 3, 4, 1
b) 1, 3, 4, 2, 5
c) 3, 1, 2, 5, 4
d) 2, 4, 1, 5, 3

18. Com relação à normalização de um banco de dados é correto afirmar que:
a) A normalização de um modelo relacional visa, principalmente, reduzir a redundância de dados aumentando a sua integridade.
b) A normalização aplica-se a um modelo entidade-relacionamento e tem como principal função a remoção de ambiguidades.
c) A maioria dos SGBDs atuais aplica automaticamente a normalização.
d) A normalização de banco de dados é necessária apenas quando se busca eficiência nas consultas aos bancos de dados relacionais.

Para as questões 19 e 20, considere a seguinte modelagem, referente ao banco de dados de um sistema de gerenciamento de eventos classificados por categoria. Para tais eventos, os ingressos podem ser comprados individualmente e nominalmente apenas por usuários cadastrados no sistema, restringindo-se a um único ingresso por usuário para cada evento cadastrado.



19. Considerando a modelagem do banco de dados anteriormente apresentada, analise as seguintes afirmativas sobre problemas existentes ou falta de boas práticas aplicadas nessa modelagem:

I. O tipo utilizado na coluna `data_hora` da tabela `Compra` não é o mais apropriado.
II. A tabela `Usuario` não deveria ter a coluna `id_compra`.
III. A coluna `Nome` da tabela `Categoria` deveria ser uma coluna da tabela `Evento`.
IV. As linhas que representam relações entre tabelas poderiam não estar se cruzando.

Estão corretas as afirmativas:
a) I, II, III
b) I, III, IV
c) II, III, IV
d) I, II, IV

20. Considerando o modelo anteriormente apresentado, analise as seguintes afirmativas:

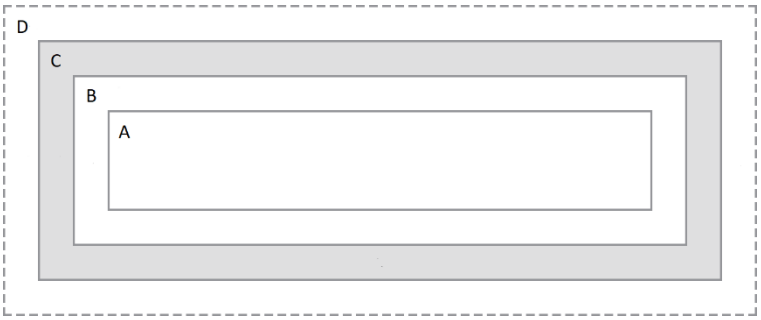
I. Para se visualizar para quais eventos cada usuário comprou ingresso, usa-se o seguinte comando SQL: `SELECT * FROM evento, categoria, compra, usuario`
II. Para se visualizar os nomes dos usuários que compraram ingresso para o evento de ID igual a 2, usamos o seguinte comando SQL:
```
SELECT usuario.nome FROM usuario
WHERE usuario.id_compra=compra.id AND compra.id_evento=2
```
III. Para se visualizar os eventos que não foram relacionados a nenhuma categoria, usamos o seguinte comando SQL:
```
SELECT evento.nome FROM evento
WHERE evento.id_categoria=0
```
IV. Para se visualizar a quantidade de eventos cadastrados no sistema, usamos o seguinte comando SQL: `SELECT SUM(id) FROM evento`

Quanto às alternativas anteriores, são falsas:
a) apenas uma
b) apenas duas
c) apenas três
d) todas

21. Durante uma viagem, um desenvolvedor está trabalhando em um projeto cujo repositório de arquivos está em um servidor que usa o Git como sistema de controle de versões. Por estar em trânsito, esse desenvolvedor encontra-se sem acesso à Internet, porém ele deseja consolidar as modificações que fez para posteriormente enviá-las para o repositório assim que tiver conexão à rede. Para realizar esse versionamento, o desenvolvedor deve:

a) realizar um *commit* local e, posteriormente, fazer o envio da nova versão para o servidor onde está hospedado o repositório.
b) fazer uma cópia dos arquivos em um diretório, renomeando esse novo diretório com número de versão maior que o atual.
c) fazer solicitação de *commit* ao servidor, de maneira que essa solicitação fique pendente e, tão logo haja conexão à rede, o código seja enviado automaticamente para o repositório.
d) criar um *branch* a partir do repositório local, colocando todos os arquivos da nova versão no diretório *trunk*.

22. A ideia do "modelo em caixa" (*box model*) do CSS (*Cascading Style Sheet*) é de representar qualquer elemento de uma página Web com uma área retangular à qual se pode adicionar bordas e fazer ajustes em termos de tamanho e espaçamento. Uma ilustração representando esse modelo seria:



Nessa representação, as letras A, B, C e D referem-se, respectivamente, a:

a) *content*, *margin*, *border* e *padding*
b) *border*, *padding*, *margin* e *content*
c) *content*, *padding*, *border* e *margin*
d) *padding*, *border*, *margin* e *content*

23. Analise as seguintes afirmativas:

I. O encapsulamento permite que uma classe defina métodos com o mesmo nome de métodos presentes em sua superclasse desde que esses métodos tenham argumentos diferentes.
II. Na linguagem de programação Java, uma instância de uma classe *C* que implementa uma interface *I* é objeto tanto do tipo definido pela interface *I* quanto do tipo definido pela classe *C*.
III. Na linguagem de programação Java, classes abstratas não precisam ser completamente abstratas, ao contrário das interfaces. Classes abstratas podem ter métodos implementados que serão herdados por suas subclasses.

A análise permite concluir que:

a) apenas as afirmativas II e III estão corretas.
b) apenas as afirmativas I e II estão corretas.
c) apenas a afirmativa II está correta.
d) apenas a afirmativa I está correta.

24. Considere os seguintes trechos de código fonte implementado na linguagem de programação Java, referentes a três classes pertencentes a um mesmo pacote:

```
public abstract class C1 {
  public abstract Object criar();
  public void exibir() {
    System.out.println("Olá, mundo");
  }
}

public class C2 extends C1 {
  static int i = 0;
  Integer j;
  public Object criar() {
    i++;
    j = new Integer(i);
    return j;
  }
  public void exibir() {
    System.out.println("j = " + j);
  }
}

public class C3 extends C1 {
  double d = 3.14;
  Float f;
  public Object criar() {
    d = d + 1.0;
    f = new Float(d);
    return f;
  }
  public void exibir() {
    System.out.println("f = " + f);
  }
}
```

```
public class Main {
  public static void main(String args[]) {
    C1 a = new C2();
    C1 b = new C2();
    C1 c = new C3();
    Object o1 = a.criar();
    o1 = a.criar();
    Object o2 = b.criar();
    Object o3 = c.criar();
    o3 = c.criar();
    a.exibir();
    b.exibir();
    c.exibir();
    System.out.print(" " + o1);
    System.out.print(" " + o2);
    System.out.print(" " + o3);
  }
}
```

Assinale a alternativa que apresenta corretamente os valores impressos pela execução do programa resultante da compilação desse conjunto de classes:

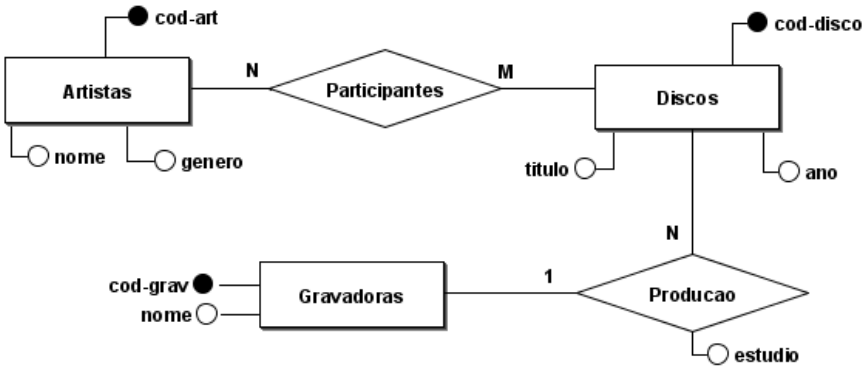
a) j = 2
   j = 1
   f = 5.14
   2 1 5.14

b) j = 2
   j = 3
   f = 5.14
   2 3 5.14

c) Olá, mundo
   Olá, mundo
   Olá, mundo
   2 1 5.14

d) Olá, mundo
   Olá, mundo
   Olá, mundo
   2 3 5.14

25. Considere o seguinte Diagrama Entidade-Relacionamento (ER) que representa um conjunto de entidades e seus relacionamentos no contexto do projeto de um banco de dados:

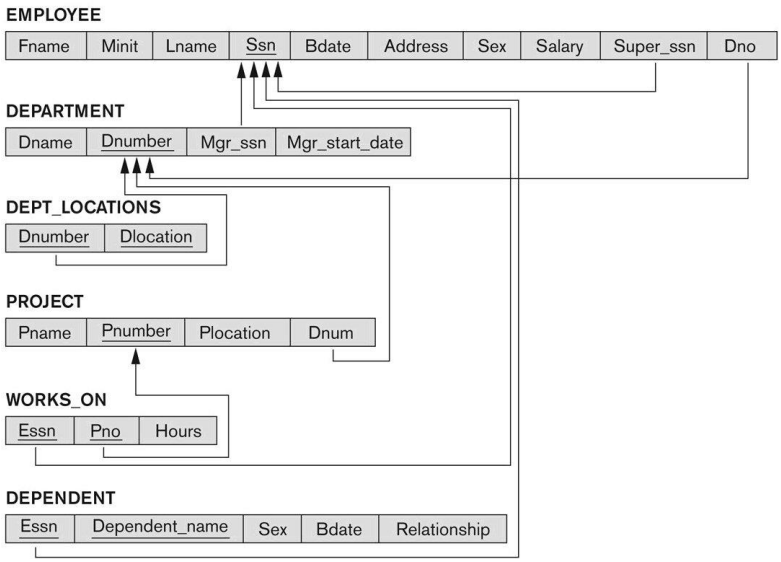


Um mapeamento desse diagrama para um esquema relacional descrevendo possíveis tabelas e seus respectivos campos (estando os campos chave sublinhados) seria:

a) ARTISTAS (cod-art, nome, genero)
   PARTICIPANTES (cod-art, cod-disco)
   DISCOS (cod-disco, titulo, ano)
   PRODUCAO (cod-grav, cod-disco, estudio)
   GRAVADORAS (cod-grav, nome)
b) ARTISTAS (cod-art, nome, genero)
   PARTICIPANTES (cod-art, cod-disco)
   DISCOS (cod-disco, titulo, ano, cod-grav, estudio)
   GRAVADORAS (cod-grav, nome)
c) ARTISTAS (cod-art, nome, genero)
   PARTICIPANTES (cod-art, cod-disco)
   DISCOS (cod-disco, titulo, ano, cod-grav, nome, estudio)
   PRODUCAO (cod-grav, cod-disco, estudio)
   GRAVADORAS (cod-grav, nome)
d) ARTISTAS (cod-art, nome, genero)
   DISCOS (cod-disco, titulo, ano, cod-art)
   PRODUCAO (cod-grav, cod-disco, estudio)
   GRAVADORAS (cod-grav, nome)

26. São métodos do ciclo de vida de uma *Activity* da plataforma Android:
a) *onCreate*, *onStart*, *onUpdate*, *onPause*, *onResume*
b) *onCreate*, *onStop*, *onSavedInstanceState*, *onService*, *onStop*
c) *onCreate*, *onStart*, *onResume*, *onBundle*, *onSavedInstanceState*
d) *onCreate*, *onStart*, *onResume*, *onPause*, *onStop*

Para as questões 27 e 28, considere o seguinte modelo relacional:



27. Qual das opções abaixo apresenta a consulta que lista os primeiros nomes de todos os gerentes que não possuem dependentes?

a) SELECT fname FROM EMPLOYEE WHERE
(ssn IN (SELECT mgr_ssn FROM DEPARTMENT)) AND
(ssn NOT IN (SELECT essn FROM DEPENDENT));

b) SELECT fname FROM EMPLOYEE WHERE
(ssn IN (SELECT mgr_ssn FROM DEPARTMENT)) OR
(ssn NOT IN (SELECT essn FROM DEPENDENT));

c) SELECT fname FROM EMPLOYEE WHERE
(ssn IN (SELECT mgr_ssn FROM DEPARTMENT)) OR
(ssn IN (NOT SELECT essn FROM DEPENDENT));

d) SELECT fname FROM EMPLOYEE WHERE
(ssn NOT IN (SELECT mgr_ssn FROM DEPARTMENT)) AND
(ssn NOT IN (SELECT essn FROM DEPENDENT));

28. Qual das opções abaixo apresenta a consulta que lista os primeiros nomes (fname) e endereços (address) de todos os empregados que trabalham em pelo menos um projeto localizado em 'Natal' (plocation) mas cujo departamento do empregado (dno) não é localizado em 'Natal' (dlocation)?

a)
```
SELECT fname, address FROM EMPLOYEE
  WHERE ((ssn IN (SELECT essn FROM WORKS_ON JOIN PROJECT ON pnumber=pno
          WHERE plocation='Natal'))
  AND
  (ssn NOT IN
      (SELECT ssn FROM EMPLOYEE WHERE dno IN
          (SELECT dnumber FROM (DEPARTMENT NATURAL JOIN DEPT_LOCATIONS)
          WHERE dlocation='Natal'))));
```

b)
```
SELECT fname, address FROM EMPLOYEE
  WHERE ((ssn NOT IN (SELECT essn FROM WORKS_ON JOIN PROJECT ON pnumber=pno
  WHERE plocation='Natal'))
  AND
  (ssn NOT IN
      (SELECT ssn FROM EMPLOYEE WHERE dno IN
      (SELECT dnumber FROM (DEPARTMENT NATURAL JOIN DEPT_LOCATIONS)
          WHERE dlocation='Natal'))));
```

c)
```
SELECT fname, address FROM EMPLOYEE
  WHERE((ssn IN (SELECT essn FROM WORKS_ON JOIN PROJECT ON pnumber=pno
  WHERE plocation='Natal'))
  AND
  (ssn NOT IN
      (SELECT ssn FROM EMPLOYEE WHERE dno NOT IN
      (SELECT dnumber FROM (DEPARTMENT NATURAL JOIN DEPT_LOCATIONS)
          WHERE dlocation='Natal'))));
```

d)
```
SELECT fname, address FROM EMPLOYEE
  WHERE((ssn IN (SELECT essn FROM WORKS_ON JOIN PROJECT ON pnumber=pno
  WHERE plocation='Natal'))
  AND
  (ssn IN (SELECT ssn FROM EMPLOYEE WHERE dno IN
      (SELECT dnumber FROM (DEPARTMENT NATURAL JOIN DEPT_LOCATIONS)
          WHERE dlocation='Natal'))));
```

29. Considerando a utilização do sistema de controle de versões Git, pode-se afirmar que:

I. Tudo no Git tem seu *checksum* (valor para verificação de integridade) calculado antes que seja armazenado e então passa a ser referenciado pelo seu *checksum*. Isso significa que é impossível mudar o conteúdo de qualquer arquivo ou diretório sem que o Git tenha conhecimento.
II. Caso o desenvolvedor esteja iniciando o monitoramento de um projeto existente com Git, ele deve navegar para o diretório desse projeto e executar o comando `git init`.
III. No Git, os arquivos podem estar em um dos três seguintes estados fundamentais: consolidado (*commited*), baixado (*downloaded*) e preparado (*staged*).
IV. Um repositório remoto pode ser clonado para o espaço local de trabalho através do comando `git clone <url>`.
V. O comando `git checkout` baixa o código mais atual da linha *master* do repositório.

Estão corretas apenas as afirmações dos itens:
a) II, III e IV
b) I, II e III
c) III, IV e V
d) I, II e IV

30. No contexto da UML (*Unified Modeling Language*), um *relacionamento* é uma ligação entre itens, podendo ser representado graficamente através de diferentes tipos de linhas. Associe os tipos de relacionamentos existentes na UML a suas respectivas descrições:

I. É um relacionamento de utilização, determinando que um item usa as informações e/ou serviços de outro item, mas não necessariamente o inverso.
II. É um relacionamento entre itens gerais e tipos mais específicos desses itens.
III. É um relacionamento estrutural que especifica objetos de um item conectados a objetos de outro item. A partir desse relacionamento, é possível navegar de um objeto de uma classe para um objeto de outra classe e vice-versa.

(a) Associação
(b) Dependência
(c) Generalização

Assinale a alternativa que contém a associação correta:
a) I-a, II-b, III-c
b) I-b, II-a, III-c
c) I-b, II-c, III-a
d) I-c, II-b, III-a